Exc No: 14

Raw sockets to implement packet sniffing

Aim:

Write a code using RAW sockets to implement packet sniffing

Algorithm:

```
import socket, struct, datetime

def mac(b): return ':'.join(f'{x:02x}' for x in b)
def ip(b): return '.'.join(map(str, b))

def parse_eth(frame):
    dst, src, proto = struct.unpack(
        '!6s6sH', frame[:14])
    return mac(dst), mac(src), socket.
    ntohs(proto), frame[14:]

def parse_ipv4(pkt):
    v_ihl = pkt[0]
    ihl = (v_ihl & 0x0F) * 4
    proto = pkt[9]
    src = ip(pkt[12:16]); dst = ip(pkt
    [16:20])
    return {'proto': proto, 'src': src, 'dst
    dst, 'payload': pkt[ihl:]}
```

```
def parse_tcp(p):
    s, d, seq, ack, off_flags = struct.
    unpack('! HHLH', p[:14])
    offset = (off_flags >> 12) * 4
    return s, d, offset, len(p[offset:])

def parse_udp(p):
    s, d, length = struct.unpack('! HHH', p[:6])
    return s, d, length-8

def parse_icmp(p):
    t, c = struct.unpack('! BB', p[:2]);
    return t, c, len(p[4:])

def main():
    try:
        s = socket.socket(socket.AF_PACKET,
socket.SOCK_RAW, socket.ntohs(0x0003))
    except PermissionError:
        print("Permission denied. Run as root");
        return
    print("Sniffing... ctrl+C to stop\n")
    try:
        while True:
            raw, _ = s.recvfrom(65535)
            dst_mac, src_mac, eth_proto,
            payload = parse_eth(raw)
            ts = datatime.datatime.now()
            strftime('%H:%M:%S:%f')[:-3]
            if eth_proto != 0x800:
                print(f"{ts} | {src_mac} ->
{dst_mac} | non-IPv4 proto=0x{
eth_proto:04x}"); continue
```

```
iph = parse_ipv4(payload)
p = iph['proto'];
src = iph['src'];
dst = iph['dst']
if p == 6:
try:
    sp, dp, offset, dlen =
parse_tcp(iph['payload'])
print(f"{ts} | TCP {src}:
{sp} -> {dst}:{dp} data={dlen}")
except: print(f"{ts} | TCP
malformed {src} -> {dst}")

if __name__ == "__main__":
    ping_server();
```

Sample input and output:

O/P:

aa:bb:cc:dd:ee:ff + 11:22:33:44:55:66

ethertype = 0x0814

len = 98

first bytes:

45 00 00 3c 1c 46 40 00 40 06 b1 e6 c0 a8 01 02 c0 a8 01

11:22:33:44:55:66 → aa:bb:cc:dd:ff

ethertype = 0x0806 len = 60

first bytes:
000108000604000 1aa bb cc
dd ee ff c0a801 02 c0a80101

de: ad: be: ef: 00: 01 → ff: ee: dd: cc:
bb: aa

ethertype = 0x0800 len = 74

first bytes:
45000003a 9a 7a 40004011 a3
fac0.a80.1 c5c0a80101

Result:
Thus packet sniffing is done using RAW sockets

13/10